# Engineering Impacts of Anonymous Author Code Review: A Field Experiment

Emerson Murphy-Hill, Jillian Dicker, Margaret Morrow Hodges, Carolyn D. Egelman, Ciera Jaspan, Lan Cheng, Elizabeth Kammer, Ben Holtz, Matthew A. Jorde, Andrea Knight Dolan, Collin Green

**Abstract**—Code review is a powerful technique to ensure high quality software and spread knowledge of best coding practices between engineers. Unfortunately, code reviewers may have biases about authors of the code they are reviewing, which can lead to inequitable experiences and outcomes. In principle, anonymous author code review can reduce the impact of such biases by withholding an author's identity from a reviewer. In this paper, to understand the engineering effects of using author anonymous code review in a practical setting, we applied the technique to 5217 code reviews performed by 300 software engineers at Google. Our results suggest that during anonymous author code review, reviewers can frequently guess authors' identities; that focus is reduced on reviewer-author power dynamics; and that the practice poses a barrier to offline, high-bandwidth conversations. Based on our findings, we recommend that those who choose to implement anonymous author code review should reveal the time zone of the author by default, have a break-the-glass option for revealing author identity, and reveal author identity directly after the review.

**Index Terms**—code review, unbiasing

## 1 Introduction

While developers believe that code changes are accepted based on change quality and fitness [1], prior research suggests that when women use profile pictures and gender-identifiable names, the acceptance of their open source code contributions drops, compared to peers with gender-neutral profiles [2]. Previous research suggests that such gender disparities are due to implicit gender bias and have been replicated in a variety of work contexts [3], and also extend beyond gender to race [4], age [5], and physical attractiveness [6].

Outside of code review, implicit bias in professional decision making is increasingly handled through anonymization, where irrelevant personal details are purposefully hidden from the decision maker. For example, research on performing orchestra auditions without seeing the person who auditioned "fostered impartiality in hiring and increased the proportion of women in symphony orchestras" [7]. Academic papers reviewed by scholars who are aware of author identity gives "a significant advantage to papers with famous authors and authors from high-prestige institutions" compared to when author identity is not revealed during review [8].

With a similar motivation, having engineers review code changes without being explicitly informed of who made those changes – *anonymous author code review*[1] – can in principle reduce the effect of bias in organizations. Indeed, in response [10] to the GitHub study on gender bias [2], Mozilla developed a browser extension that anonymizes GitHub pull requests [13], yet the extension has not been evaluated. In the social sciences, Kim and colleagues identified anonymous author code review as a step towards reducing structural sexism [9], a problem tech companies such as Google have been criticized for [14].

But we know essentially nothing about how anonymous author code review would work in practice. After Facebook reportedly replicated the GitHub study [15], news reports indicate that Facebook's VP of Engineering rejected the practice saying "Hiding the identity of authors or reviewers is counterproductive from an engineering perspective" [16]. But is it counterproductive in practice? Unfortunately, we know of no empirical evidence.

So while in principle anonymous author code review reduces the impact of biases, we don't know what other effects it might have on the engineering process. This paper seeks to understand these effects by answering the following research questions:

**RQ1: How often can reviewers guess author identities?** When reviewers are aware of who the author is during anonymous author review, this undermines the effectiveness of anonymous author review. Analogously, prior research suggests that during double-blind paper academic paper review, reviewers can rarely guess author identities [17], [18]. We also investigate whether reviewers are less able to guess authors during *certain types of reviews*, making these types more amenable to anonymous author code review.

**RQ2: How does anonymous author code review change reviewers' velocity?** Speed of code review is a significant concern across companies and organizations that practice it [19], yet anonymous author code review may slow down the code review process, for example, when reviewers can't easily contact authors for high-bandwidth communications.

**RQ3: How does anonymous author code review change review quality?** Prior research suggests that double-blind research paper reviews are of equal [20], [21], [22], [23] or higher quality [24], [25] as single-blind reviews because reviewers are more critical when they are unaware of author identity [26]. We may expect similar quality effects for anony-

1. Others [9], [10] have called this technique 'blind' code review, alluding to 'blind reviewing' of scientific articles. We avoid this term purposefully because 'single-blind reviewing' of scientific papers means that the author is not aware of reviewer's identity, whereas here we mean the reviewer is not aware of the author's identity. Some have also argued the metaphorical use of the word 'blind' in these contexts is ableist [11], [12].

mous author code review.

**RQ4: What effect does anonymous author code review have on reviewers' and authors' perceptions of fairness?** Fairness is both considered important by software engineers [27] and a central goal of anonymous author code review. While our field study is not well-suited to study fairness objectively, we take the approach of prior work [24], [28] to study it though the subjective perceptions of reviewers and authors.

**RQ5: What do engineers perceive as the biggest advantages and disadvantages of anonymous author code review?** We next explore the tradeoffs involved in performing anonymous author code review, beyond the effects explored in the prior questions.

**RQ6: What features are important to an implementation of anonymous author code review?** Understanding what features of an anonymous author code review tool is designed to help organizations decide how they might implement anonymous author code review.

In answering these questions, this paper contributes the first empirical study of anonymous author code review. While the motivation for this work is reducing bias during code review, in this paper we will not directly examine whether bias is actually reduced, which would likely require a larger-scale study than we performed here.

## 2 Background: Code Review at Google

**Process.** At Google, most of our code resides in one large repository. When an engineer wants to make a change, they:

- Create a changelist (CL) that contains diffs of one or more files, similar to pull requests on GitHub.
- Use the tool that facilitates this review process – Critique – that has the ability to display diffs, post and reply to comments about a CL, and display analysis results, such as code coverage and linter warnings.
- Choose one or more appropriate reviewers either manually or by getting a recommendation from Critique. If the author is not an owner of the code being changed, a reviewer must be an owner. The majority of CLs are reviewed by someone on the author's team.
- Tell Critique to notify reviewers to begin their review. Reviewers add comments about the change and ask for further changes, if necessary.
- Make fixes and respond to comments about the change. The process of asking for changes and making changes may be repeated several times.
- Merge the changelist into the repository, once the requisite positive signal – a "looks good to me" or LGTM – is granted by reviewers.

More information about the review process at Google can be found in prior work [19].

**Readability.** Google has instituted a mandatory coding style and recommended best practices for each of the various languages in wide use, such as Java, C/C++, and Python. Additionally, for each language there is a process by which an engineer can demonstrate their knowledge of the best practices and agreement to enforce them. This process is known as readability. If a changelist author modifies a file in a language for which they do not have readability, the changelist must be approved by a reviewer with readability in that language. Linter tools catch most style violations automatically. Review for readability is primarily intended to ensure best practices are being followed, for example, in naming conventions, recommended APIs, modular design, and good testing practices.

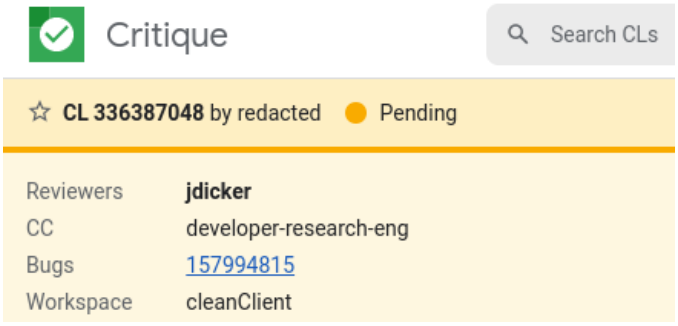



Fig. 1: What a reviewer (jdicker) using our extension sees at the top of Critique when reviewing.

**Comparison to Other Code Review Processes.** Sadowski and colleagues have compared the code review process with Critique at Google to code review processes elsewhere [19] using Rigby and Bird's convergent practices framework, which compares code review across multiple organizations [29]. Sadowski and colleagues conclude that Critique at Google is similar to other code review contexts insofar as it's a lightweight and flexible code review process, but the notion of explicit ownership and readability is unique. We also note that Critique supports asynchronous review with email notifications like other systems, but in contrast to systems like AMD's CodeCollaborator [30] and Microsoft's CodeFlow [31], Critique does not explicitly support live chatting between the author and reviewers.

## 3 Method

To answer our research questions, in early 2019, we built a browser extension that automatically hides information about the authors of changes from reviewers (Section 3.1), deployed the extension to volunteering developers at Google (Section 3.2), and collected (Section 3.3) and analyzed (Section 3.4) qualitative and quantitative data on code review experience and outcomes.

### 3.1 Implementation of Anonymous Author Code Review

While changing Critique would be the most seamless way to implement anonymous author code review at Google, a browser extension allowed us to evaluate the idea of author anonymous code reviews without any changes to critical engineering infrastructure. The extension we built removed authorship information from Critique, anywhere the author's username appears; from our email client (Gmail), in Critique emails where the author's username or profile images appear; and in CL Monitor, another browser extension used by many engineers at Google that notifies them of incoming and outgoing reviews. Our browser extension also prompted participants for information about each CL they reviewed, directly after they provided LGTM. Figure 1 shows a screenshot of how our extension removes the author's username from

Critique. While the reviewer experience in Critique changes, authors see Critique as normal, with all usernames included.

In taking the browser extension approach, we were unable to remove author information from the following sources:

- Emails and notifications from a users' mobile device,
- Emails and notifications on non-Google-owned devices, and
- Bugs that link to the CL being reviewed.

Due to these limitations, we had to ask participants to create an email filter where Critique emails would skip their inbox and to refrain from looking at such emails on mobile and non-corp devices. We also told participants that our extension would not be able to remove author information in all contexts and to work around such issues. Participant instructions can be found in the Supplemental Material Sections 1 through 3.

Finally, we anticipated that in some situations, participants would need to know an author's identity, so the extension had a button to deanonymize a CL in Critique.

### 3.2 Participants

We recruited participants (Table 1) by sampling engineers across Google who met the following criteria:

- Employed by Google for at least 6 months, in an attempt to avoid conflating the experience of anonymous author code review with overall novelty of Google's review process;
- Been on the same team for at least 3 months, in an attempt avoid conflating the experience of anonymous author code review with the novelty of being on a new team; and
- Reviewed at least 10 CLs in the two weeks before the study, in an attempt to ensure that the participant would perform enough reviews during the study period to generate a sufficient volume of data.

Additionally, using stratified random sampling we aimed to have a subsample of participants – 25% – be readability reviewers. When a readability review is requested, the readability reviewer is assigned randomly from a queue of already-certified developers, a queue that contains hundreds of potential reviewers for popular languages like Java or C/C++. Thus, we hypothesize that for such readability reviews, reviewers are unlikely to know the identity of authors. If that hypothesis is true, readability reviews are thus good candidates for anonymous author code review. To recruit readability reviewers, for this subsample we required engineers to have performed at least 2 readability reviews per week in the two weeks prior to the study.

One third of all volunteers were randomly assigned to a control group (stratifying on readability), which enabled us to compare data collected from engineers reviewing with and without anonymous author code review. The control group performed reviews in a standard, author-visible manner, but completed reports similar to those in the treatment (that is, anonymous author) group. A post-hoc analysis shows that the random assignment to control and treatment groups yielded statistically similar levels of seniority (median level 4, one level above entry-level, $p = 0.59$), though the control group had statistically significantly longer tenure (median 3.67 years for control versus 3.63 years for control), using a Wilcoxon rank sum test ($p < .001$). This motivates the use of covariate controls in our analysis (Section 3.4).

| | | | | |
|---|---|---|---|---|
| Engineers Invited | 1650 (350R) | | | |
| Engineers Volunteered | 439 (128R) | | | |
| | **Treatment** | | **Control** | |
| Engineers Enrolled | 330 | (96R) | 109 | (32R) |
| Engineers Reporting on 1+ CLs | 300 | (90R) | 95 | (29R) |
| Submitted CL Reports | 5217 | (1858R) | 1935 | (682R) |
| Submitted Final Reports | 282 | (85R) | 92 | (29R) |

TABLE 1: Statistics about who participated and the number of reports they submitted. Numbers in parentheses indicate readability. Example: 439 engineers volunteered, of which 128 had readability. Example: 5217 total CLs were reviewed by the treatment group, of which 1858 were by readability reviewers.

The study complied with human Google ethics guidelines for conducting research with human participants and underwent internal employee privacy review. Participants could terminate their participation in the study at any time.

### 3.3 Data Collection

#### *3.3.1 Quantitative Data*

Quantitatively, we used metrics from several sources. First, we logged when participants pressed the deanonymize button to answer RQ1, in part. Second, we used metrics derived from tool logs to collect active reviewing time as a measure of the reviewers' velocity (RQ2). Active reviewing time is the time a reviewer spends actively viewing, commenting, or working on a changelist, which may include time outside of Critique in other tools, such as time spent looking up APIs or documentation. Further information on the active reviewing time metric can be found in our prior work [32]. Third, we use quantitative metrics to measure which changelists were rolled back in the 10 months after the code reviews took place. We posit that the higher the quality of the code review (RQ3), the lower the likelihood of rollback.

#### *3.3.2 Qualitative Data*

Qualitative data was collected to through two report types completed by participants:

- **CL Reports.** Our browser extension asked participating reviewers to fill out a report immediately after every LGTM, asking the participating reviewer to guess the author's identity (RQ1, treatment group), and what perceived effect the anonymization (or identity, for the control group) had on review velocity (RQ2), quality (RQ3), and fairness (RQ4).
- **Final Report.** Sent by email at the end of study period, a report asking participants about the reviewer experience during the study period, including review velocity (RQ2), quality (RQ3), and fairness (RQ4).[2] Questions also asked participants about overall advantages and disadvantages of anonymous author code review (RQ5) and desired features of an anonymous author code review system (RQ6).

2. We additionally asked *authors* of CLs reviewed as part of the study (both treatment and control) about their perceptions about fairness of the review they received (RQ4).

Table 1 indicates how the number of reports participants filled out. The texts of each question will be described with results (Section 4), and blank reports are available in the Supplemental Material, Sections 4 through 9.

### 3.4 Data Analysis

Analyses varied from research question to research question, but all statistical analyses were performed in R. R scripts were code reviewed by both an experienced software engineer and a quantitative analyst. Statistical tests were run with an alpha value of .05 to determine significant effects.

In every inferential statistical analyses in this paper, we used one of two types of regressions: review-level regressions (RLR) and participant-level regressions (PLR). Regressions allowed us to isolate effects of interest by controlling for covariates; for example, to estimate whether review time was different between groups, we controlled for changelist size and the seniority of the reviewer, among other variables listed below. Both RLRs and PLRs included the following fixed effects:

- *Group*: Whether the participant was in the control or treatment group.
- *Participating reviewer variables*: tenure (years at Google), seniority (level), role (individual contributor vs. tech lead, etc.), job code (software engineer vs research engineer, site reliability engineer, etc), region (US West vs Latin America, US South, etc), and whether the participant was a readability reviewer (binary). We controlled for tenure, seniority, and readability because they mediate code review pushback [33]; role and job code because different types of developers have different code review motivations and expectations [31]; and region because culture influences engineering practice [34].

RLRs included a random effect (reviewer identity) to account for individual variation from reviewer to reviewer. RLRs included these additional fixed effects:

- *Author variables*: The same variables as the participating reviewer (above), but for the author of the change (e.g., the tenure of the author).
- *Changelist variables*: the log of the number of reviewers, the CL size[3], whether the reviewer was a readability reviewer, and whether the CL changed "code". Here code means that at least one file is changed that's attributed to one of the known 40 coding languages at Google, including C++, Java, and Go. Examples of non-coding changes are those that exclusively change documentation, access control, and build management. Additionally, we included a binary "large-scale change" variable; these are relatively low-risk changes to a broad swath of our monolithic codebase (e.g. changing all uses of one API method to another). Large-scale changes are typically split up into multiple smaller CLs, where each CL is sent to appropriate code owners for review. We controlled for number of reviewers, size, and readability because they mediate code review pushback [33]; controlled for code changes because programming language correlates with pull request acceptance [2]; and controlled for large-scale changes because they are fundamentally different than other types of changes.
- *Relationship*: We modeled past interactions between reviewer and author as 'insider', 'outsider', or an 'unclear' relationship. An insider relationship means that the reviewer has reviewed 10 or more CLs at Google for the author prior to the CL in question; an outsider relationship means the reviewer has reviewed 2 or fewer CLs for this author previously; and unclear otherwise. We included this control variable because a similar notion of relationship in open source software correlates with pull request acceptance [2]

To convey a sense of overall model fit, we report adjusted $R^2$ values for Ordinary Least Squares regressions and adjusted McFadden's pseudo $R^2$ [35] for other regressions.

3. The categories of code review sizes are: XS (0-9 lines changed), S (10-49 lines changed), M (50-249 lines changed), L (250-999 lines changed), XL (over 1,000 lines changed), and U (could not be calculated).

## 4 Results

This section is structured as follows. In the next section, we describe to what extent participants and the CLs they reported on are representative of Google's engineering population. Then, starting in Section 4.2, we report on the answers to our research questions.

### 4.1 Participation and Representativeness

Invited engineers could decide to participate or not, and if they participated, decide whether to report on an individual changelist or not. This section reports on the extent to which those engineers who participated and those changelists that were reported on were representative.

#### 4.1.1 Who Was Likely to Participate and Not Participate

To determine whether some invited engineers were more or less likely to volunteer in our study, we created a logistic PLR that predicts participation (adjusted McFadden pseudo-$R^2 = 0.18$). We found:

- Engineers with readability were twice as likely to participate as those without (Odds Ratio $= 2.0$, $p < .001$).
- Engineers who have been at Google for 7 or more years were less likely to participate than engineers who have been at Google for less than a year (OR $= 0.33$, $p < .001$).
- Engineers in the US South were more likely to participate than in the US West region, with 4 out of 6 invitees volunteering in the US South (OR $= 6.8$, $p = .042$).

No other predictors emerged as statistically significant.

#### 4.1.2 Which Changelists Were Reported On

While we instructed participating engineers to report on their experience reviewing every CL during the study period, sometimes they did not. We ran a logistic RLR that predicted whether a CL was reported on (adjusted McFadden pseudo-$R^2 = 0.09$). While the raw median percentage of CLs reported on by control group participants was 88%, and the median

| | Normal reviews | | Readability reviews | |
|---|---|---|---|---|
| Correct Guess | 76.6% | (n=3239) | 4.6% | (n=24) |
| Uncertain | 21.3% | (n=903) | 95.1% | (n=500) |
| Incorrect Guess | 2.1% | (n=88) | 0.4% | (n=2) |

TABLE 2: How often participants were able to correctly guess the identity of the authors of the changelists they reviewed.

was 84% for treatment group participants, this difference was not statistically significant ($p = 0.8$), controlling for other covariates. However, a variety of other covariates were associated with an increase in the odds that a CL was reported on (e.g. CLs from outsiders and readability reviews), while other covariates were associated with a decrease in those odds (e.g. more reviewers and large scale changes) These findings underscore the importance of using RLR regressions, which control for these covariates, in the remainder of this paper.

Finally, during the study, we asked participants to what extent the CLs they reviewed during the study were typical of their personal experience. The response distribution between the two groups was similar, but the treatment group was slightly more likely to perceive the CLs they reviewed as being typical. 71% of the control group rated this as "Very typical", versus 77% of the treatment group. "Somewhat typical" was chosen by 25% of the control group versus 22% of the treatment group, and "Not at all typical" was chosen by 5% and 1%, respectively.

#### 4.1.3 Participating Reviewers Per Changelist

In the changelists that were part of this study, most were reviewed by just one (55% of reviews) or two (29% of reviews) reviewers. 0.5% of reviews had multiple study participants as reviewers. 0.15% of reviews had a treatment group and a control group reviewer, so cross-contamination of results was a limited risk.

### 4.2 RQ1: How often can reviewers guess author identities?

Reviewers may know the identity of authors without being explicitly informed. In this section, we investigate how often reviewers with anonymous authors can guess author identity, by what means, and for what reasons.

#### 4.2.1 Author Guessability Rates

To examine how often authors were guessable, we asked treatment group participants to guess author usernames after saying they were either "Very Certain", "Somewhat certain", or "Uncertain" of author identity. If they chose "Uncertain", we did not ask them to provide a guess. We performed two data cleaning steps: we manually inspected and fixed incorrectly-spelled guesses, then removed any remaining guesses that didn't correspond to any known Google engineer (e.g, blank responses). We next took the remaining cases and categorized them into correct guesses and incorrect guesses. Results are displayed in Table 2. Let's first examine readability reviews, since we hypothesized that authors and reviewers were unlikely to know each other during these reviews because reviewers are assigned randomly. The data confirms this – in 95% of anonymous author readability reviews, the reviewer reported being uncertain of the author's identity. Of the cases where the readability reviewer did know the author's identity (n=24), the most common reasons were the author contacting the reviewer outside of code review (n=8, 33%) and the part of the codebase being changed (n=6, 25%). With non-readability reviews, in 77% of cases reviewers guessed the author correctly. In 21% of cases the reviewers were explicitly uncertain of the author's identity. In 2% of cases, the authors were at least somewhat certain of the author's identity, but guessed incorrectly.

#### 4.2.2 Author Guessability Mechanisms

We asked treatment group participants how they knew author identities. These participants had a set of 11 predefined options, but could also state their own reasons; participants chose to do so in 660 cases. One author of this paper coded respondents' reasons and grouped them into several reason categories. Because reviewers could choose multiple reasons, the total number of reviews sums to more than the total number of reports submitted.

The most common reason that reviewers with anonymous authors knew the identity of authors was for two contextual reasons (n=2845, 53%). The first was that authors could determine the author from the part of the codebase that was being changed. The second was from the nature of the change, such as the language or programming style. As one participant stated, "The author has been doing a large number of similar migration CLs".

The second most common reason was because the author and reviewer communicated (n=1294, 24%). This was usually because the author contacted the reviewer. One participant gave an example as "The author pinged our chat about fixing a test". This also occurred when the author and reviewer collaborated prior to a change or could guess identity by virtue of being on the same team. For instance, one participant noted "I know who is working on this specific change from sprint planning/standup".

The third most common reason was because of some limitation of our implementation of anonymous author code review (n=579, 11%). This was usually because the description of the change indicated – either explicitly or implicitly – the author's identity. For instance, the "design doc linked in the CL description" revealed author identity.

The fourth most common reason was through deductive means (n=259, 5%), typically because a change did or didn't need approval from an owner of the code being changed. This is because at Google, at least one reviewer must be an owner if the author is not an owner. Thus, if Critique indicates that an owner's review is not required, then the author must be an owner, which narrows the set of potential authors.

Other less common reasons include from the knowing what tasks coworkers are performing prior to the review (n=161, 3%), because some part of our tool failed (n=147, 3%), because the reviewer decided to deanonymize the author (n=43, 1%), and because the reviewer was in close enough physical proximity to happen to see the CL on the author's screen (n=7, <1%).

Some of the less frequent reasons for author deanonymization during anonymous author code review can be alleviated in a straightforward way (e.g. redacting bug assignee when browsing a linked bug). However, the top two most common reasons – the part of the codebase being changed and the

nature of the change – appear more challenging to alleviate. The next most common set of reasons – author to reviewer communication – may be alleviated in part by building anonymous communication, such as through anonymous messaging. If anonymous messaging between an author and reviewers were retained alongside a CL, side benefits might accrue, such as the retention of design rationales contained in author-reviewer discussions.

#### 4.2.3 Why Reviewers Need to Know Author Identities

When treatment group participants filled out a report on their experience after a code review during the study, our chrome extension inserted logs about whether the participant explicitly requested to deanonymize the CL. In total, 47 reports with logs indicated that the participant requested deanonymization, compared to 5112 reports with logs where the participant did not.

To determine why this rare event occurred, we asked participants "If you [deanonymized] the CL using the extension, why did you do so?" We qualitatively coded the 53[4] open ended responses into 10 types of reasons, with some responses containing multiple reasons.

- A set of common deanonymization reasons were about the context of the CL. For example, one participant stated, they were "curious about the reason of the change." In n=10 (19%) cases, the reviewer needed additional context to review the change that wasn't contained in the original CL. In n=5 (9%) cases, the reviewer may have needed to supply additional context about the change to the author, depending on the author's identity. In n=8 (15%) cases, the author needed to be made aware of other people relevant to this change, and the reviewer did not know if the author is one of these people. Similarly, in n=3 (6%) cases, the reviewer needed to assess the level of knowledge the author has about the context.
- In n=7 (13%) cases, the reviewer needed to know whether the author had implicit permission to make a change, including changes to access control lists (ACLs) and TODOs. As one participant stated, "CL author added themselves to OWNERS file. Diff showed addition as "redacted." I needed to know who is being added to the OWNERS file."
- In n=6 (11%) cases, the reviewer mentioned deanonymization because they needed to contact the author.
- In n=5 (9%) cases, our extension inadvertently redacted a relevant URL in Critique, so deanonymization was required to click through the URL.
- In n=2 (4%) cases, the reviewer deanonymized to determine who made the change so that they could thank the author.
- In n=2 (4%) cases, the reviewer indicated simply being curious about the author's identity.
- In n=13 (25%) cases, it was unclear from the reports why the reviewer needed to deanonymize the CL.

As the list above suggests, out-of-Critique communication between reviewers and authors occurs for some CLs. In the post-LGTM report, we asked respondents, "If you contacted the author or the author contacted you outside of Critique, why?" We received 1009 responses to this question.

Authors were about four times more likely to contact reviewers than reviewers contacting authors. Although it was sometimes difficult to tell who initiated contact, based on our reading of the responses, in 59% percent of cases authors contacted reviewers; in 14% of cases reviewers contacted authors, and in 27% of cases it was unclear.

It was clear that a substantial number of responses were largely about how the reviewer knew the author's identity, rather than about the *reason* behind the contact. Including only answers to the latter (n=691), several themes emerged:

- The most common theme (n=211, 31%) was that the author contacted the reviewer at the start of the review to introduce the CL, request a review, offer some up-front context, or ask for a swift review due to the urgency of the CL. For example, one participant noted, "The author promised me to deliver a fix for a tool. Later I saw a fix for the said tool. Inferred the author's identity."
- The second most common theme (n=185, 27%) was making contact to discuss a CL or address/clarify a comment made during the review. For instance, one participant commented, "To ask follow-up questions on the comments I've left in Critique."
- The third most common theme (n=107, 15%) was to ask the author or reviewer a question related to the review. As one participant noted, "He asked me questions about. . . the tests he was going to change, as he wasn't sure of how they worked."
- Other reasons include the author or reviewer mentioning their preference for a conversation outside of Critique due to the relative speed or ease as compared to going back and forth in comments (n=45, 7%); the author contacting the reviewer to remind them about the CL awaiting review (n=35, 5%); contact made convenient by close physical proximity of author and reviewers (n=33, 5%); expediting a CL rollback (n=15, 2%); mentioning the change fixing something that's broken or breaking something (n=11, 2%); asking for clarification around the readability process (n=10, 1%); and needing permission to access a shared resource (n=6, 1%).

### 4.3 RQ2: How does anonymous author code review change reviewers' velocity?

We measured the effect on review velocity through both qualitative and quantitative measures.

During the course of the study, we asked the following after a participant LGTM'd a CL:

4. The six-report discrepancy between the 47 logs indicating deanonymization and 53 open ended answers indicating deanonymization could be due to either our extension failing to log events correctly or to participants misinterpreting the question. Manual inspection of the open ended responses suggests a likely combination of these two. Similarly, the discrepancy between the 47 logs indicating deanonymization and the 43 reports where participants said they knew authors' identity because they pressed the deanonymize button, may be explained by four engineers either forgetting that they deanonymized the review or forgetting author identity after they deanonymized.

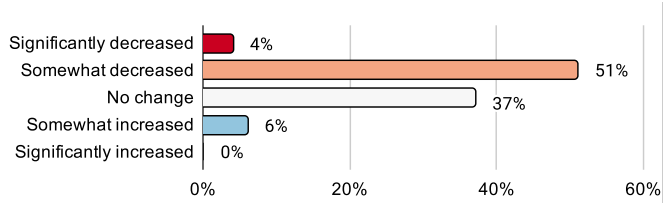



Fig. 2: Treatment participants' expectations about what effect anonymous author code review would have on review velocity if practiced regularly at Google. An additional 2% of participants responded "I don't know".

- Treatment participants were asked "what effect did the [anonymous author] code review process had on reviewing velocity, compared to [non-anonymous] code review?" For 89% of CLs, participants said anonymous author code review did not change velocity; for 6% anonymous author review had a negative impact on velocity; for 2% a positive impact; and the rest were not known.
- Conversely, we asked control participants to estimate the effect of knowing the identity of the author compared to not knowing their identity in terms of review velocity. Here, for 69% of CLs the reviewers said knowing the identity had no effect; for 3% knowing identity decreased velocity; for 26% knowing identity increased velocity.

After the study, we asked treatment participants "what [effect on velocity] did the [anonymous author] code review process have, compared to [non-anonymous] code review?" 59% said anonymous author code review had no effect on velocity; 5% said a positive effect on velocity, 36% said a negative effect, and the remainder did not know.

We also wanted to know the impact of anonymous author code review on velocity more broadly, so we asked all participants, "If [anonymous author] code review was regularly practiced at Google, I expect that my engineering velocity would be...", followed by a rating. As shown in Figure 2, most (55%) treatment group participants thought that anonymous author code review would have a negative effect on review velocity, with most of the remainder of participants (37%) expecting no effect.

Overall, this data suggests that **participants perceived that anonymous author code review generally has a negative or neutral effect on code review velocity**.

Although perceived review time is important, we can also measure active reviewing time as an objective measure of velocity. To do so, we compared not only the control and treatment groups during the study, but also those groups against their own review velocities pre- and post-study. Thus, we separated reviews into those performed in the approximately two weeks during the study, in the two-week period before the study began, and in the two-week period after the study ended. We defined study beginning and ending on a per-participant basis, since participants may have begun using our chrome extension or filled in the final report at any point after we invited them to do so. We defined the study beginning for a participant as the time when the participant made the first comment on a CL that they filed a report for.

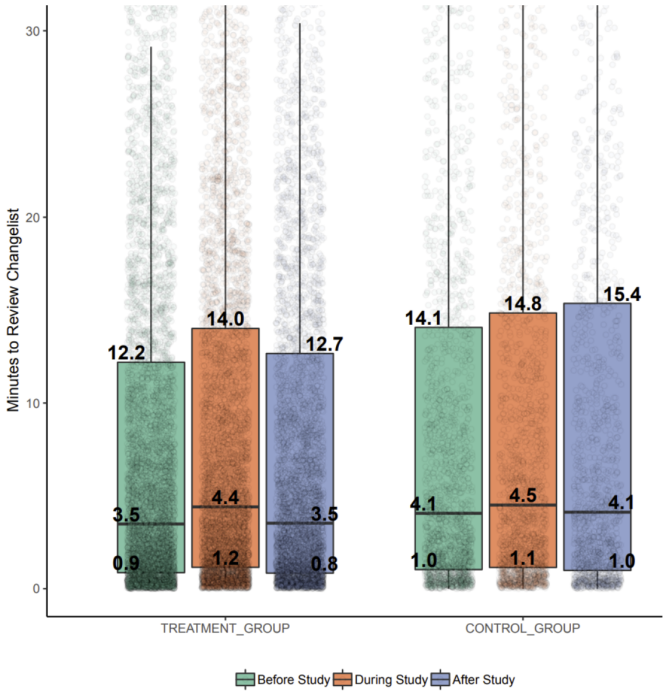



Fig. 3: The time participants spent actively reviewing each changelist in minutes, comparing treatment and control groups in terms of review time before (left), during (center), and after the study (right).

We defined the study ending for a participant as the time when the participant submitted their last report.

Figure 3 illustrates raw reviewing time differences between the control group and treatment group, and in the pre-study ($n_{control} = 2351$ and $n_{treatment} = 7612$ changelists), during-study ($n_{control} = 2169$ and $n_{treatment} = 6308$ changelists), and post-study ($n_{control} = 1702$ and $n_{treatment} = 6149$ changelists) periods. In the figure, we show boxplots that summarize reviewing time distributions, overlaid with individual code review times as circles. We do not display any code reviews that took more than about 30 minutes, but these are accounted for by the boxplots.

This raw data suggests that reviewing time increased during the study period for both control and treatment groups – from 3.5 to 4.4 minutes for the treatment group, and from 4.1 to 4.5 for the control group. So was review time lengthened by anonymous author review? To examine this, we created a linear RLR predicting log review time (adjusted McFadden pseudo-$R^2 = 0.07$). We use log review time since review time is highly skewed (most reviews take less than 5 minutes, but some reviews can take 30 minutes or more). In addition to the existing linear RLR covariates, we add a three-level fixed-effect for time period (before, during, and after the study).

This RLR indicates that, indeed, changelist review time was longer during the study than before the study (by 10%, $p < .001$) and after the study (by 6%, $p < .001$). However, this RLR also indicates that the time taken to review in the treatment (anonymous author) group was not statistically significantly different than the time taken in the control group ($p = .346$).

Why might both the control and treatment group show an increase in reviewing time during the study? Potential expla-

nations include both groups of participants being influenced by modulating behavior simply by being aware of identity-issues and both groups changing their behavior due to being part of a study [36].[5]

In sum, because both control and treatment groups reviewed more slowly during the study to a statistically similar extent, our results suggest that **anonymous author code review did not substantially change objective reviewing velocity during the study, compared to non-anonymous review, in our study setting.**

This conclusion applies to anonymous author code review in practice at companies like Google, that is, when author identities can often be guessed (Section 4.2). Other contexts may exist where guessability is lower, such as open source projects that accept pull requests from a wide variety of external contributors; in these contexts, would we expect different reviewing velocities? We answer this question by comparing normal treatment group reviews where the reviewer knew the author identity to those where the author's identity was not known. A detailed description of this analysis is in the Supplemental Material Section 12; in short, we find that there exists fundamental differences in code reviews when the author is guessable compared to non-guessable. Therefore, comparing guessable to non-guessable author code reviews on metrics like velocity or quality is not a reasonable comparison, so we make no further attempt to do so in the remainder of the paper.

### 4.4 RQ3: How does anonymous author code review change review quality?

Given the effect of code review on software quality [37], [38], [39], [40], we asked how participants perceived their review quality during and after the study.

After participants LGTM'd a CL:

- We asked treatment participants to rate what effect the anonymous author code review process had on code quality, compared to non-anonymous code review. For 92% of CLs, participants said anonymous author code review had no impact on quality; for 1% anonymous author review had a negative impact on quality; for 4% a positive impact; and the rest were not known.
- Conversely, we asked control participants to estimate the effect of knowing the identity of the author compared to not knowing their identity in terms of review quality. Here, for 79% of CLs the reviewers said knowing the identity had no effect; for 7% knowing identity decreased quality; for 12% knowing identity increased quality.

After the study, we asked treatment participants "what [effect on quality] did the [anonymous author] code review process have, compared to [non-anonymous] code review?" 75% said anonymous author code review had no effect on quality; 18% said a positive effect on quality, 7% said a negative effect, and the remainder did not know.

We examined the impact of anonymous author code review on quality more broadly, so we asked all participants, "If anonymous author code review was regularly practiced at Google, I expect that..." As Figure 4 shows, most (57%) treatment group participants thought that anonymous author code review would be neutral toward review quality, with most of the remainder of the treatment group participants (24%) expecting positive impact on quality.

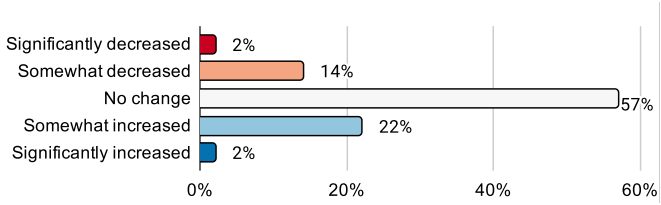



Fig. 4: Treatment participants' expectations about what effect anonymous author code review would have on review quality if practiced regularly at Google. An additional 1% of participants responded "I don't know".

Finally, we ran a logistic RLR to predict whether the changelists reviewed by treatment participants were more or less likely to be rolled back in the roughly 17 month period after the experiment took place, one measure of review quality (adjusted McFadden pseudo-$R^2$ = 0.0008). We found no statistically significant difference in the odds of rollbacks, comparing the treatment CLs reviewed during the study period compared to the CLs in the two week period before the study ($p = .71$) and the two week period after the study ($p = .46$). However, surprisingly, control group participants' changelists saw a slight but statistically significant rise in the odds of a rollback for CLs reviewed during the study period compared to the pre-period (Odds Ratio 1.02, $p < .001$) and post-period (OR 1.01, $p < .001$). This evidence suggests that review quality is at least as good during anonymous author code review as during non-anonymous review.

Overall, this data suggests that **anonymous author code review generally has a neutral effect on code review quality.**

### 4.5 RQ4: What effect does anonymous author code review have on reviewers' and authors' perceptions of fairness?

At its core, the purpose of anonymous author code review is to allow engineers to review code without being burdened by any biases they might have about the author. Thus, we examined whether code reviewers (Section 4.5.1) and authors (Section 4.5.2) were able to perceive any differences in how reviewers treated authors during anonymous author code reviews, specifically from a fairness perspective.

#### *4.5.1 Reviewer Fairness Perceptions*

We first asked reviewers whether they perceived any differences in their own fairness while performing anonymous author code reviews. After participants LGTM'd a CL:

- We asked treatment participants to rate what effect the anonymous author code review process had on reviewing fairness, compared to non-anonymous code review. For 93% of CLs, participants said anonymous author code review did not change fairness; for 4%

5. The increase is not due to filling out the post-LGTM report, as review time does not include the time to fill out this form.

anonymous author review made the review more fair; for <1%, less fair; and the rest were not known.
- Conversely, we asked control participants to estimate the effect of knowing the identity of the author compared to not knowing their identity in terms of review fairness. Here, for 89% of CLs the reviewers said knowing the identity had no effect; for 3% knowing identity decreased fairness; for 2% knowing identity increased fairness; and the rest were not known.

After the study, we asked treatment reviewers "what [effect on fairness] did the anonymous author code review process have, compared to [non-anonymous] code review?" 66% said anonymous author code review had no effect on fairness; 31% said a positive effect on fairness, 1% said a negative effect, and the remainder did not know.

Overall, this data suggests that **most reviewers perceived that anonymous author code review has a neutral or positive effect on code review fairness**.

#### *4.5.2 Author Fairness Perceptions*

We also asked code review authors if they perceived any fairness differences. Note that we don't necessarily expect an increase or decrease in perceived fairness. On one hand, if authors perceived previous non-anonymous reviews as unfair, then they may perceive anonymous author review as more fair than they expected. On the other hand, authors who previously had been treated with undue deference might perceive anonymous author reviews as less fair than they expected.

After reviews were completed by study participants, we asked authors of those CLs "During code review for this changelist, did you experience being treated as fairly as you expected?" We asked authors about the following CLs:

- All control group CLs. These represent CLs when the reviewer is aware of author identity.
- Treatment group CLs where the reviewer indicated that they were "Uncertain" of the author's identity. These represent CLs where the reviewer did not have knowledge of the author's identity.

We also surveyed authors once, and only once, to reduce survey fatigue. We did not indicate in this author survey that the reviewers were participating in an experiment, nor whether they were reviewing with anonymous authors (treatment group) or non-anonymously (control group). Moreover, study participants were discouraged from discussing the study with other Google engineers, so as to avoid biasing authors.

We separately analyze reviews where the study participant was the readability reviewer. Table 3 shows authors' fairness ratings.

The 'Normal reviews' columns shows that for most reviews in both control and treatment groups, authors did not feel treated more or less fairly than they expected (93% of control group authors, 94% of treatment group authors). The control and treatment group were not significantly different ($p = .97$), according to a linear RLR (adjusted McFadden pseudo-$R^2 = -2.6$) that predicts perceived fairness, where "less fair" is coded as -1, "more fair" is coded as 1, and otherwise as 0.

The 'Readability reviews' columns tell a curious story; control group participants' readability reviews were perceived as more fair than those from the treatment group. The interaction between control/treatment group and readability is statistically significant ($p = .002$), according to the same RLR.

Since we were surprised by this finding, we evaluated the hypothesis that the control group participants gave inordinately fair reviews.[6] We asked authors who received readability reviews from reviewers not invited to be part of the study to report the fairness of the reviews that they received. Results are shown in the 'Post Study' column in Table 3.

These readability CLs reviewed by non-participants show levels of author-perceived fairness slightly closer to the treatment group than the control group. An Ordinary Least Squares regression with the same fixed effects as an RLR predicting perceived fairness (adjusted $R^2 = .16$) provides more definitive evidence of this: non-participant CLs are significantly different than control group CLs ($p < .001$) but not significantly different from treatment group CLs ($p = .38$), in terms of author-perceived fairness.[7] In other words, control group CLs appear to be the outliers here. The hypothesis that the control group gave fairer reviews than the treatment group – rather than the treatment group giving particularly unfair ones – is supported.

Overall, this data suggests that **authors who unknowingly received anonymous author reviews did not perceive a significant difference in fairness, compared to authors who received non-anonymous reviews.**

### 4.6 RQ5: What do engineers perceive as the biggest advantages and disadvantages of anonymous author code review?

#### *4.6.1 Observed Benefits.*

We asked treatment group participants "What was the main benefit you experienced during this study while performing [anonymous author] code review?" One author qualitatively coded each of the 198 responses into 10 emergent categories.

The definitions of these categories are:

- *No personal benefits / not sure.* The most commonly mentioned benefit was none that was perceptible to the participant (n=71, 36%). To quote one participant, "It is hard for me to find a clear benefit as I nearly always knew who I was reviewing (due to the change being made)."
- *More thorough review without reliance on author identity.* The second most commonly mentioned benefit was that reviews were more thorough (n=65, 33%), because the reviewer could not rely on a high level of author expertise and consequently conducted their reviews in a more neutral way, independent of the authority of the author, the relationship between the reviewer and the author, or trust based on prior experiences with the author. Many comments mentioned the necessity of taking a closer look without being able to assume the author possessed certain knowledge, as well

6. Our analyses do not support three alternate hypotheses; see Supplementary Material, Section 13.

7. A caveat of this model is requires the omission of the reviewer random effect, because all non-participant CLs were reviewed by different reviewers. Use of a fixed-effect model here requires us to relax the assumption of independence.

| | Normal reviews | | Readability reviews | | |
|---|---|---|---|---|---|
| | Treatment | Control | Treatment | Control | Post Study |
| More fairly than expected | 5.6% (n=23) | 6.6% (n=29) | 6.5% (n=17) | 17.1% (n=24) | 9.5% (n=33) |
| About the same / don't know / n/a | 93.6% (n=383) | 92.7% (n=406) | 92.0% (n=242) | 82.9% (n=116) | 89.7% (n=312) |
| Less fairly than expected | 0.7% (n=3) | 0.7% (n=3) | 1.5% (n=4) | 0% (n=0) | 0.9% (n=3) |

TABLE 3: How authors of changelists perceived fairness of normal and readability reviews from developers in the control group, the treatment group, and a set of non-participants after the study.

as feeling more empowered to leave feedback without regard to whether it was appropriate given the author's status. To quote one participant, "When I didn't know for sure that an author was a subject matter expert on a CL, it made me pay more attention to the content than I would have otherwise."

- *Reduce bias.* Another commonly mentioned benefit was a reduction in bias towards the author (n=45, 23%). A specifically mentioned category of reducing bias was *reducing level/tenure bias* (n=8, 4%). Note that there was a high overlap between the benefits of "reduce bias" and "more thorough review without reliance on author identity," and the latter benefit can be considered a form of reduced bias. During analysis, only comments that explicitly mentioned the terms bias and fairness or the concept of treating all CLs equally were tagged with the theme "reduce bias". To quote one respondent, "I think that I (and I would daresay most reviewers) probably apply some bias to CLs based on various attributes we attach to the author (tech level, tenure at Google, readability, role, past interactions or discussions with the authors, etc.). In some these cases, the biases serve as a bit of a short-circuit and I'm more willing to give an LGTM to someone that I "trust" more. . . even if I don't fully understand the content of the change."
- *Other benefits.* Several participants mentioned experiencing other benefits, including the review process being quicker or simpler when they didn't have to consider additional context related to the author's identity (n=20, 10%); special value for readability reviews where reviewers and authors have little relationship (n=17, 9%); enjoying and reflecting on a different kind of review process (n=12, 6%); needing to provide better documentation and context in a CL (n=9, 5%); encouraging promptness (n=4, 2%); and removing CL notifications from email (n=3, 2%).[8]

#### 4.6.2 Observed Drawbacks.

As with benefits, we also asked treatment group participants to state the main drawback they observed during the study of anonymous author code review. In the 233 responses, the most commonly observed themes were:

- *Lack of context to aid in decisions.* This refers to additional information that can be gleaned from knowing an author's identity, which reviewers stated would have aided in a swifter or more tailored review (n=111, 48%). Examples of such contextual information include the background and motivation behind the CL, the time zone of the author, the author's familiarity with the part of the codebase being modified, and the CL's level of urgency. One participant said, "Often, CLs come in with context that was chatted about offline or over other channels (bugs, emails) and knowing who authored the CL helps trigger the context of the CL."
- *Barriers to offline communication.* Similarly, this category refers to the need for reviewers and authors to communicate outside of a CL to quickly convey information (n=48, 21%). One participant said, "At times, it is useful to have an in person conversation to clarify a point or to make a design decision more quickly than trading back and forth over critique. [Anonymous author code review] made that process a little bit harder."
- *Minimal or no impact/often knew author.* This category describes minor or no effects of anonymous author code review, often because the reviewer could guess the author (n=45, 19%). One participant said, "I'm able to identify the author for most of the CLs anyway, making this experience less efficient to me."
- *Reduced velocity.* Some participants reported that their reviews took longer when reviewing with anonymous authors (n=35, 15%). As one participant said, "Because of the increased fairness and quality, reviews took longer."
- *Inconvenience/performance issues with extension or reports.* This theme refers to the limitations of the study that were drawbacks (n=30, 13%), such as the extension redacting many usernames in error or slowing down Gmail inordinately. One participant said, "Not really a drawback of the process, per se, but the aggressiveness with which the Chrome extension redacted names sometimes made it hard to get additional information". A similar theme was *lack of email notifications*, which was a limitation required by our chrome extension (n=13, 6%).
- *Other drawbacks.* Participants reported other drawbacks, including reviewers being unable to tailor feedback to the background of the author, such as giving detailed explanations to new employees (n=21, 9%); not being able to distinguish core contributors from external contributors (n=18, 8%); not being sure or stating non-drawbacks (n=12, 5%); reducing personal

8. "Removed CLs from the inbox" was a requirement of the study that participants should have Critique emails bypass their inbox. This would not be a constraint in a full implementation of anonymous author code review.

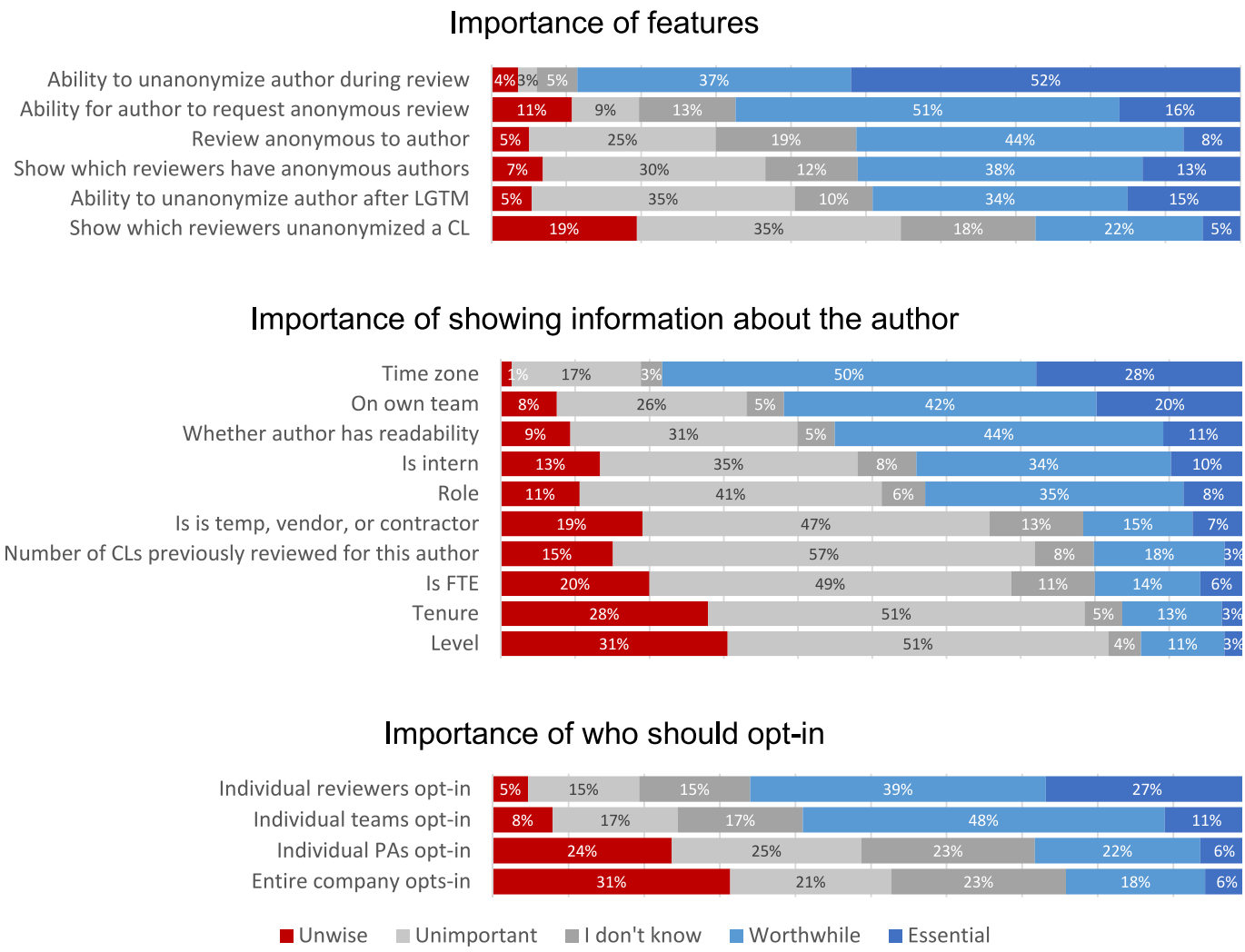



Fig. 5: How participants rated the importance of various potential features of an anonymous author code review system.

interactions with peers (n=12, 5%); being less knowledgeable about colleagues' work (n=11, 5%); challenges in looking back over old CLs without seeing author names (n=8, 3%); awkwardness around the code review process (n=7, 3%); and difficulty in prioritizing important CLs over less important ones (n=6, 3%).

### 4.7 RQ6: What features are important to an implementation of anonymous author code review?

At the end of the study, we asked participants about a set of features that anonymous author code review might have, if implemented at Google. The top of Figure 5 shows a variety of features that we asked participants to rate the importance of. Of these features, participants were most strongly positive about the ability to deanonymize author the author during review, with 52% of participants noting the feature as essential. Most respondents rated the following features as at least worthwhile: the ability of the author to request anonymous author review, making the reviewer anonymous to author identity when no specific reviewer is required (e.g. when using a reviewer queue), showing which reviewers are reviewing with anonymous authors, and deanonymizing the author after granting LGTM. Showing which reviewers deanonymized a CL was rated positively by some respondents (27%), but also was rated as unwise by 19% of participants.

We also asked about what information should be revealed about the author, if the author's identity were anonymized, as shown in the middle of Figure 5. This figure indicates a variety of opinions, overall it indicates that most participants support revealing author time zones, whether the author is on the reviewer's team, and whether the author has readability. Author tenure and level were the two types of information that participants were more likely to believe would be unwise to show than worthwhile or essential.

We also asked "Suppose [an anonymous author] review system is opt-in, where those who opt-in perform [anonymous author] review by default, but reviewers can [deanonymize] as they see fit. How important is it to allow opt-in to be chosen by each of the following?" Results at the bottom of Figure 5. The results indicate participants were positive about having individual reviewers and teams opt-in, mixed about whether product areas (PAs) opt-in, and somewhat negative about having the entire company opt-in.

## 5 Limitations

Readers should consider several limitations of this study while interpreting its results.

- Results obtained from a similar study, run in another organization or an open source software project, may differ.
- Although we randomly selected participants, those who opted to participate likely do not represent the full population of developers, both within Google and beyond.
- Participants used anonymous author code review over a relatively short period of time. As we describe in Section 6, the effect of anonymous author code review may differ if practiced over a longer period.
- Participants' self-reports are likely influenced by cognitive biases and by question wording. For instance, one question asked authors to rate fairness relative to their expectations as a baseline and another asked reviewers to rate fairness relative to a hypothetical counterfactual baseline. Both baselines makes interpreting the answer challenging.
- While we reported the number of times various themes from our qualitative data were mentioned, these numbers may not capture the frequency or severity. Rather, they likely capture how salient or memorable the theme was for participants.
- Given the number of open-ended responses and the amount of work required, we opted to have just one coder categorize the open-ended themes. More robust themes may have emerged with more coders.
- While this paper examined the effects of anonymous author code review in several dimensions, the practice may yet have other effects not fully explored in this study, effects such as relationship building and knowledge sharing.

## 6 Discussion

We found that for about 77% of non-readability reviews, developers performing anonymous author review correctly guessed the author's identity. Rather than concluding that blind code review doesn't work or is impossible in practice, we instead argue that guessability is simply the unavoidable reality of contemporary code review. Nonetheless, our results suggest that "low context" changes are less guessable than other changes, and thus may be particularly effective places for implementing anonymous author review; such low context changes include readability reviews, large scale changes, and small changes. But even for reviews where authors are guessable, there may yet be benefits to anonymous author code review. First, as McKinley argues of double blind paper review, "the very act of omitting author details on the paper. . . reminds reviewers that they should judge the paper on its merits rather than based on whomever they guess the authors might be" [41]. Second, we hypothesize that if author information is regularly hidden during code review, it may become less prominent in reviewers' minds (this was not the case in this study, since we frequently prompted developers to guess author identity). Third, we hypothesize that even if reviewers are 95% certain of author identity, the remaining 5% of doubt will be enough to change one's behavior. These hypotheses could be evaluated in future studies.

One finding from the study was that code reviews took longer, for both the treatment group and the control group. While we interpret this to mean that anonymous author code review *per se* did not cause the increase in reviewing time, it is still possible that anonymous author code review could nonetheless increase reviewing time in a non-experimental field deployment. If the time increase is due to a novelty effect [42], we would expect active review time to come back down after the novelty wears off. On the other hand, anonymous author code reviewers might take more time to review because the nature of their feedback changes, for example by being more explicit, which may in turn yield higher quality reviews. Which of these possibilities will materialize in practice requires further study in a larger scale but less experimental (e.g., no reports) study.

We found that fairness was, for most reviews, not perceptibly different from reviewers' and authors' expectations. We do not interpret this to mean that anonymous author code review cannot make reviews more fair, but rather that any changes in fairness were not perceptible to the group of respondents as a whole. We instead argue that anonymous author code review is more fair by construction, because irrelevant personal details are excluded, to the extent possible, from the decision making process.

For organizations that choose to implement anonymous author code review, we have several recommendations based on our results. The first is that the downsides appear minor for implementing author-anonymous review on low-context changes, where the reviewer has limited contextual information about the change prior to review. In open source, for instance, pull requests from newcomers might be a low-risk place to start implementing anonymous author code review. In terms of which features to implement for anonymous author code review, we recommend:

- Implement a break-the-glass option for revealing author identity, which participants were strongly in favor of. The downsides to this option are minimal; our results suggest the feature is rarely used, and when it is, it's used largely for important reasons such as understanding who's making access control changes.
- Implement displaying author time zone information, which participants were also strongly in favor of. This allows developers to make informed decisions about when to review.
- Reveal author information by default after LGTM is granted. This would allow developers to maintain familiarity with their colleagues' work, which was a downside of anonymous author code review as implemented in this study.

## 7 Related Work

Studies on the impact of code review have become increasingly popular. Microsoft [31] found that developers report using code review not only for finding bugs, but also for knowledge transfer. Developers reported that they provided more useful and detailed feedback when they were also experts in the code being reviewed, as they had relevant context for the review. Sadowski and colleagues also found that at Google,

there were similar practices around using code review for knowledge transfer, and that reviewers with context provided valuable feedback to the author [19]. A later study at Microsoft confirmed that a reviewer provided more useful comments when they had prior experience with the code being reviewed, though there was no difference based on whether the reviewer and author were on the same team [43]. This is particularly important for the effectiveness of anonymous author code review: reviewers provide more useful comments if they are familiar with the code, but our study suggests that reviewers who are familiar with the code may be more likely to successfully guess the author's identity.

Prior work has shown that in open-source code reviews, female authors are less likely to get their patch accepted when their gender is known [2]. Additionally, German and colleagues have found that fairness is a concern in open-source code reviews [27]. German and colleagues describe four types of fairness relevant to code review: distributive fairness, procedural fairness, interaction fairness, and information fairness. Anonymous author code review addresses distributive fairness by improving the equity and equality in how authors are treated in code review. Anonymous author code review also adds procedural fairness as it is a form of bias suppression.

While there is little prior work on anonymous author code reviews, there is significant prior work in anonymization in the peer-reviewed research papers. Prior work has found that anonymizing the author and affiliations from the peer reviewers increases representation of female authors [44] and also increases the representation of less-famous authors and authors from lower-prestige institutions [8]. These results have also famously held in areas outside of peer reviewed publications, such as in orchestral auditions [7] and in hiring [45].

There has been some prior work in non-technical factors that impact the outcome of code reviews. Baysal and colleagues examined several such factors, including the author's prior experience [46]. They found that prior experience improved the likelihood that a patch is accepted, and reduced the time spent in code review. However, it is not known whether this is a direct result of the author's prior experience or an artifact of bias on the part of the reviewer. Kononenko and colleagues also examined the impact of an author's prior experience on the quality of the review and found that it does not affect review quality [47].

## 8 Conclusion

While in principle anonymization reduces bias during code review by removing decision-irrelevant information (e.g., when an engineer has gender biases, removing identity removes the most salient gender signal), the principle is threatened by practical considerations, such as the ability of reviewers to guess author identity. In this paper we described a field study of anonymous author code review, building an understanding of its benefits and drawbacks. Building on this increased understanding, we encourage researchers to investigate whether anonymous author code review reduces disparities in code review outcomes, such as the gender gap in pull request acceptance rate [2].

## Acknowledgements

Thanks to all engineers who volunteered, without whom this study would not have been possible. Thanks also to Danny Berlin, Ash Kumar, Ambar Murillo, and Rachel Potvin.

## References

[1] G. Gousios, A. Zaidman, M.-A. Storey, and A. van Deursen, "Work practices and challenges in pull-based development: The integrator's perspective," in *Proceedings of the 37th International Conference on Software Engineering*, ser. ICSE, vol. 1, May 2015, pp. 358–368. [Online]. Available: /pub/pullreqs-integrators.pdf

[2] J. Terrell, A. Kofink, J. Middleton, C. Rainear, E. Murphy-Hill, C. Parnin, and J. Stallings, "Gender differences and bias in open source: Pull request acceptance of women versus men," *PeerJ Computer Science*, vol. 3, p. e111, 2017.

[3] H. K. Davison and M. J. Burke, "Sex discrimination in simulated employment contexts: A meta-analytic investigation," *Journal of Vocational Behavior*, vol. 56, no. 2, pp. 225–248, 2000.

[4] J. H. Greenhaus, S. Parasuraman, and W. M. Wormley, "Effects of race on organizational experiences, job performance evaluations, and career outcomes," *Academy of management Journal*, vol. 33, no. 1, pp. 64–86, 1990.

[5] R. A. Gordon and R. D. Arvey, "Age bias in laboratory and field settings: A meta-analytic investigation 1," *Journal of applied social psychology*, vol. 34, no. 3, pp. 468–492, 2004.

[6] M. Hosoda, E. F. Stone-Romero, and G. Coats, "The effects of physical attractiveness on job-related outcomes: A meta-analysis of experimental studies," *Personnel psychology*, vol. 56, no. 2, pp. 431–462, 2003.

[7] C. Goldin and C. Rouse, "Orchestrating impartiality: The impact of "blind" auditions on female musicians," *American Economic Review*, vol. 90, no. 4, pp. 715–741, September 2000. [Online]. Available: http://www.aeaweb.org/articles?id=10.1257/aer.90.4.715

[8] A. Tomkins, M. Zhang, and W. D. Heavlin, "Reviewer bias in single-versus double-blind peer review," *Proceedings of the National Academy of Sciences*, vol. 114, no. 48, pp. 12 708–12 713, 2017.

[9] J. Y. Kim, G. M. Fitzsimons, and A. C. Kay, "Lean in messages increase attributions of women's responsibility for gender inequality." *Journal of personality and social psychology*, vol. 115, no. 6, p. 974, 2018.

[10] D. Marti, "Blind code reviews experiment," Available from https://blog.zgp.org/blind-code-reviews-experiment/, 2017.

[11] S. Schalk, "Metaphorically speaking: Ableist metaphors in feminist writing," *Disability Studies Quarterly*, vol. 33, no. 4, 2013.

[12] E. A. Largent and R. T. Snodgrass, "Blind peer review by academic journals," *Blinding as a solution to bias: strengthening biomedical science, forensic science, and law*, pp. 75–95, 2016.

[13] E. Humphries, "Web Extension for Debiasing Code Reviews in Splinter Experiment," Available from https://bugzilla.mozilla.org/show_bug.cgi?id=1366429, 2017.

[14] A. Nelson, "Google And The Structural Sexism Of The American Workplace," Available from https://www.forbes.com/sites/amynelson1/2018/10/30/google-and-the-structural-sexism-of-the-american-workplace, 2018.

[15] D. Seetharaman, "Facebook's Female Engineers Claim Gender Bias," https://www.wsj.com/articles/facebooks-female-engineers-claim-gender-bias-1493737116.

[16] J. C. Wong, "Facebook: leaking info about gender bias damages our 'recruiting brand'," https://www.theguardian.com/technology/2017/may/02/facebook-gender-bias-female-engineers-code.

[17] S. J. Ceci and D. Peters, "How blind is blind review?" *American Psychologist*, vol. 39, no. 12, p. 1491, 1984.

[18] C. L. Goues, Y. Brun, S. Apel, E. Berger, S. Khurshid, and Y. Smaragdakis, "Effectiveness of anonymization in double-blind review," *Communications of the ACM*, vol. 61, no. 6, pp. 30–33, 2018.

[19] C. Sadowski, E. Söderberg, L. Church, M. Sipko, and A. Bacchelli, "Modern code review: a case study at google," in *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*. ACM, 2018, pp. 181–190.

[20] M. Alam, N. Kim, J. Havey, A. Rademaker, D. Ratner, B. Tregre, D. P. West, and W. Coleman III, "Blinded vs. unblinded peer review of manuscripts submitted to a dermatology journal: a randomized multi-rater study," *British Journal of Dermatology*, vol. 165, no. 3, pp. 563–567, 2011.

[21] M. Fisher, S. B. Friedman, and B. Strauss, "The effects of blinding on acceptance of research papers by peer review," *Jama*, vol. 272, no. 2, pp. 143–146, 1994.

[22] A. C. Justice, M. K. Cho, M. A. Winker, J. A. Berlin, D. Rennie, P. Investigators *et al.*, "Does masking author identity improve peer review quality?: A randomized controlled trial," *Jama*, vol. 280, no. 3, pp. 240–242, 1998.

[23] S. van Rooyen, F. Godlee, S. Evans, R. Smith, and N. Black, "Effect of blinding and unmasking on the quality of peer review: a randomized trial," *Jama*, vol. 280, no. 3, pp. 234–237, 1998.

[24] R. A. McNutt, A. T. Evans, R. H. Fletcher, and S. W. Fletcher, "The effects of blinding on the quality of peer review: a randomized trial," *Jama*, vol. 263, no. 10, pp. 1371–1376, 1990.

[25] K. Okike, K. T. Hug, M. S. Kocher, and S. S. Leopold, "Single-blind vs double-blind peer review in the setting of author prestige," *Jama*, vol. 316, no. 12, pp. 1315–1316, 2016.

[26] R. M. Blank, "The effects of double-blind versus single-blind reviewing: Experimental evidence from the american economic review," *The American Economic Review*, pp. 1041–1067, 1991.

[27] D. German, G. Robles, G. Poo-Caamaño, X. Yang, H. Iida, and K. Inoue, ""was my contribution fairly reviewed?" a framework to study the perception of fairness in modern code reviews," in *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*. IEEE, 2018, pp. 523–534.

[28] M. Ware, "Peer review in scholarly journals: Perspective of the scholarly community–results from an international study," *Information Services & Use*, vol. 28, no. 2, pp. 109–112, 2008.

[29] P. C. Rigby and C. Bird, "Convergent software peer review practices," in *FSE*, 2013.

[30] J. Ratcliffe, "Moving software quality upstream: The positive impact of lightweight peer code review," in *Proceedings of the Pacific Northwest Software Quality Conference*, 2009.

[31] A. Bacchelli and C. Bird, "Expectations, outcomes, and challenges of modern code review," in *ICSE*, 2013.

[32] C. Jaspan, M. Jorde, C. D. Egelman, C. Green, B. Holtz, E. K. Smith, M. M. Hodges, A. Knight, E. Kammer, J. Dicker *et al.*, "Enabling the study of software development behavior with cross-tool logs," *IEEE Software*, 2020.

[33] C. Egelman, E. Murphy-Hill, L. Kammer, M. M. Hodges, C. Green, C. Jaspan, and J. Lin, "Predicting developersâĂŹ negative feelings about code review," in *International Conference on Software Engineering*, 2020.

[34] H. Shah, N. J. Nersessian, M. J. Harrold, and W. Newstetter, "Studying the influence of culture in global software engineering: thinking in terms of cultural models," in *Proceedings of the 4th international conference on Intercultural Collaboration*, 2012, pp. 77–86.

[35] D. McFadden, *Conditional logit analysis of qualitative choice behavior*, P. Zarembka, Ed. Academic Press, 1974.

[36] J. G. Adair, "The hawthorne effect: a reconsideration of the methodological artifact." *Journal of applied psychology*, vol. 69, no. 2, p. 334, 1984.

[37] S. McIntosh, Y. Kamei, B. Adams, and A. E. Hassan, "An empirical study of the impact of modern code review practices on software quality," *Empirical Software Engineering*, vol. 21, no. 5, pp. 2146–2189, 2016.

[38] P. Thongtanunam, S. McIntosh, A. E. Hassan, and H. Iida, "Revisiting code ownership and its relationship with software quality in the scope of modern code review," in *ICSE*, 2016.

[39] G. Bavota and B. Russo, "Four eyes are better than two: On the impact of code reviews on software quality," in *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2015, pp. 81–90.

[40] P. Thongtanunam, S. McIntosh, A. E. Hassan, and H. Iida, "Investigating code review practices in defective files: An empirical study of the qt system," in *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 2015, pp. 168–179.

[41] K. S. McKinley, "Improving publication quality by reducing bias with double-blind reviewing and author response," *ACM Sigplan Notices*, vol. 43, no. 8, pp. 5–9, 2008.

[42] J. D. Wells, D. E. Campbell, J. S. Valacich, and M. Featherman, "The effect of perceived novelty on the adoption of information technology innovations: a risk/reward perspective," *Decision Sciences*, vol. 41, no. 4, pp. 813–843, 2010.

[43] A. Bosu, M. Greiler, and C. Bird, "Characteristics of useful code reviews: an empirical study at microsoft," in *MSR*, 2015.

[44] A. E. Budden, T. Tregenza, L. W. Aarssen, J. Koricheva, R. Leimu, and C. J. Lortie, "Double-blind review favours increased representation of female authors," *Trends in ecology & evolution*, vol. 23, no. 1, pp. 4–6, 2008.

[45] O. Åslund and O. N. Skans, "Do anonymous job application procedures level the playing field?" *ILR Review*, vol. 65, no. 1, pp. 82–107, 2012.

[46] O. Baysal, O. Kononenko, R. Holmes, and M. W. Godfrey, "The influence of non-technical factors on code review," in *WCRE*, 2013.

[47] O. Kononenko, O. Baysal, L. Guerrouj, Y. Cao, and M. W. Godfrey, "Investigating code review quality: Do people and participation matter?" in *2015 IEEE international conference on software maintenance and evolution (ICSME)*. IEEE, 2015, pp. 111–120.

# Supplemental Material

## 1. Invitation Email

Dear Googler,

We're writing to invite you to participate in a study about "blind" code review, which we define as the practice of reviewing a changelist (CL) without explicit knowledge of the author's identity. You were selected randomly from a pool of qualified Googlers. This invitation is for you only; please do not forward it to your colleagues.

To experiment with this idea, we've created a Chrome extension that hides CL author information from Critique and Gmail, and modified CL Monitor to hide author information as well. If the experiment goes well, we'll work towards implementing the idea more widely within Google. But we need your help, by participating in this study!

To be eligible to participate, you must agree to do the following over the course of the study (just FYI on what you can expect -- no need to do these yet):

- Install our Chrome extension. The extension modifies the Critique and Gmail pages you look at to hide author information about CLs. The extension also collects information about which CL you are reviewing and what extension features you use.
- Complete a short questionnaire after each LGTM you grant.
- Complete a questionnaire at the end of the study.
- Create an email filter that collects Critique emails and skips your inbox, so that you don't get email notifications in any device, such as your mobile device (we will provide explicit instructions for doing this). If you still want to get notifications when you receive a CL to review, we recommend that you install CL Monitor. You should refrain from looking at these Critique messages on mobile and non-corp devices, since our extension will not be able to blind these messages on such devices.
- Refrain from using Inbox during the study. The reason is that our extension does not blind Critique emails in Inbox. As a reminder, Inbox will shutting down in the middle of March.
- Refrain from discussing your participation in the study with other Googlers, until the study period is complete. In particular, authors of CLs that you review will not be aware that you are participating in this experiment, and your identity will not be blinded to the author of the CL. We will, however, contact some CL authors with a survey so that they can report if they noticed any differences in how their CLs are reviewed.

Other things to note about the study:

- The study will last about two weeks
- You will be able to “unblind” any changelist in Critique, if you decide that viewing changelists in a blind manner will significantly diminish your ability to work effectively. Please use this ability sparingly and using your best judgement. You will have the opportunity to report such unblinding in the survey.
- Not every participant in the study will be performing blind code reviews. Some participants will review CLs as normal, but still be asked to answer our survey questions. These participants are critical to establish baseline measures.
- This study only applies to code reviews through Critique, not those through GitHub, Gerrit, etc.
- The purpose of the study is not to evaluate people or even the way we have implemented it with a browser extension, but instead to evaluate the impact of blind code review at Google, how/if ever we might implement it in the future.

That’s it! If you’d like to participate in this study, please fill out this short questionnaire by XXX. Once you submit the form, expect to hear from us within the next two days for further instructions on how to participate.

Sincerely,

Emerson Murphy-Hill & Jill Dicker
Engineering Productivity Research

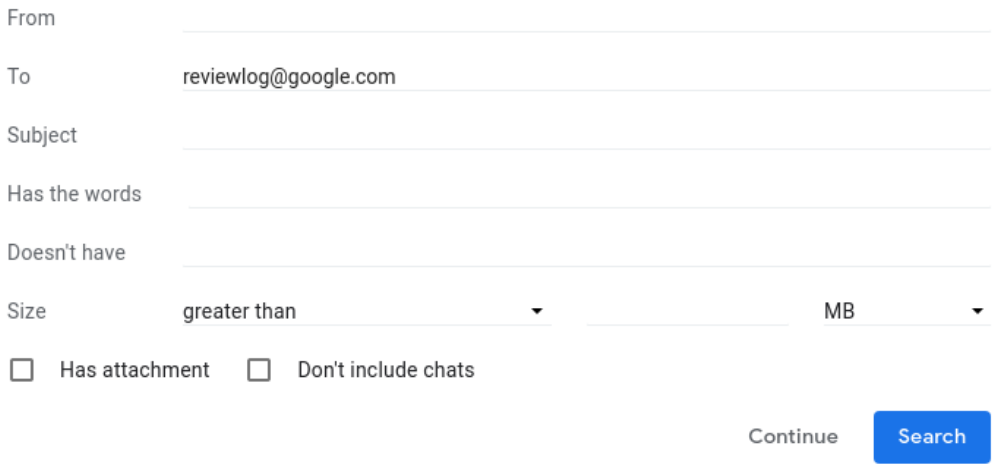
## 2. Treatment Group Welcome Email

Thank you for volunteering to be part of our study on blind code review! We hope that you are as excited as we are!

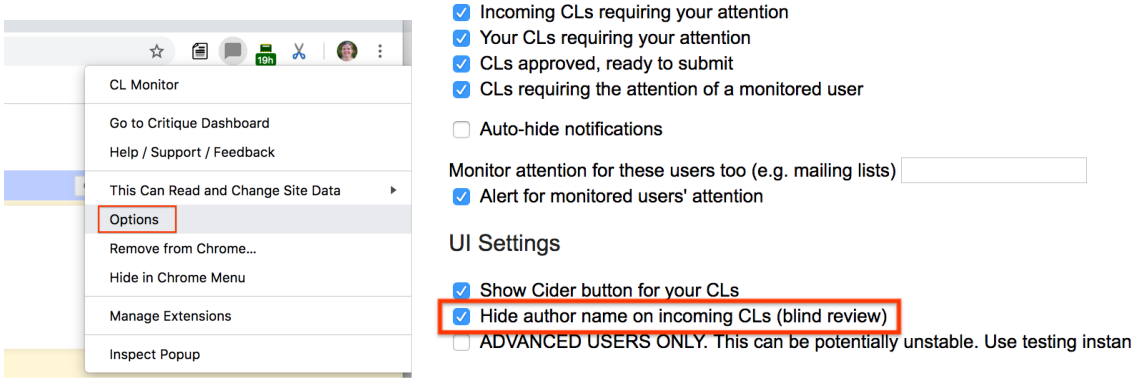
You have been selected to perform code reviews in Critique in a blind way (that is, without knowing the identity of the authors). Please complete the following instructions as soon as possible:

- Please install our Chrome extension. This extension will ensure that you receive survey invitations after each code review, and that author information is hidden from you.
- Please create an email filter that collects Critique emails and skips your inbox, so that you don’t get email notifications in any device, such as your mobile device. If you still want to get notifications when you receive a CL to review, we recommend that you install CL Monitor. To make things easier, there are two ways you can create this filter:
  A. Click on Settings -> Filters in Gmail -> Import filters, then upload this xml file; or

B. Create a filter in Gmail that skips the inbox with the following values:
to: XXX@google.com




- If you use CL Monitor, please update your extension and in the options page check the blinding option. Since CL Monitor uses local storage for preferences, you will need to check this option on your workstation *and* laptop. This will ensure that author information will be hidden in CL Monitor.




For the next two weeks:

- The extension will prompt you to fill out a survey after you grant an LGTM to a changelist. Please fill out each survey at your earliest convenience, preferably before moving on to your next task. If you're a readability reviewer, please fill out this survey before you fill out your usual readability survey.

- You will be able to “unblind” any changelist in Critique, if you decide that viewing changelists in a blind manner will significantly diminish your ability to work effectively. Please use this ability sparingly and using your best judgement. You will have the opportunity to report such unblinding in each follow-up survey.
- Refrain from looking at Critique messages on mobile and non-corp devices, since the extension will not be present on those devices to remove author information.
- Refrain from discussing your participation in the study with other Googlers, until the study period is complete (in about 2 weeks). In particular, authors of CLs that you review will not be aware that you are participating in this experiment, and your identity will not be blinded to the author of the CL. We will, however, contact some CL authors with a survey so that they can report if they noticed any differences in how their CLs are reviewed.
- Refrain from using Inbox during the study. The reason is that our extension does not blind Critique emails in Inbox. As a reminder, Inbox will be shutting down any day now.
- Note that our Chrome extension isn’t perfect, and might inadvertently reveal author information. Please do your best to work around such issues, and feel free to report bugs.

In two weeks, we will send you another email informing you that the study has ended and asking you to fill out the final survey.

Thanks again,

Emerson Murphy-Hill & Jill Dicker
Engineering Productivity Research

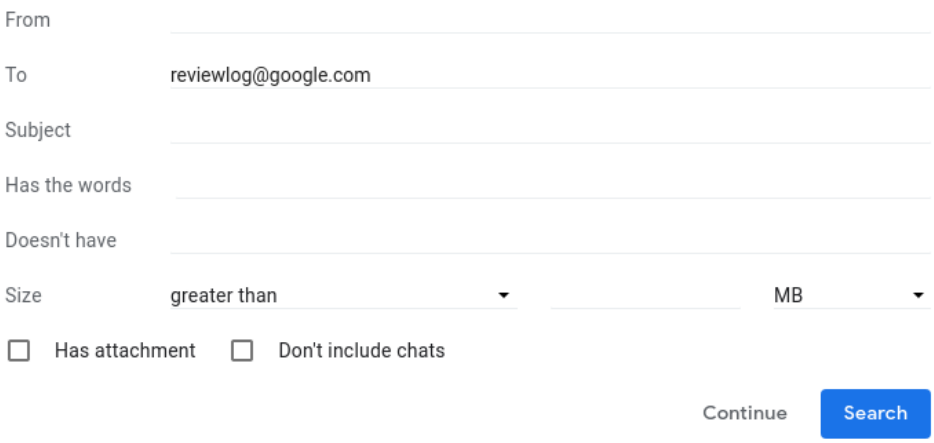
## 3. Control Group Welcome Email

Thank you for volunteering to be part of our study on blind code review! We hope that you are as excited as we are!

You have been randomly selected to be part of the control group. This means that you will be performing code reviews in Critique as you usually do (knowing the identity of authors), but will still be asked to report your experience after each code review and at the end of the study. The control group is critically important as a reference point against which to compare those who are in the treatment (blind code review) group.

Please complete the following instructions as soon as possible:

- Please install our Chrome extension. This extension will ensure that you receive survey invitations after each code review.
- Please create an email filter that collects Critique emails and skips your inbox, so that you don't get email notifications in any device, such as your mobile device. If you still want to get notifications when you receive a CL to review, we recommend that you install CL Monitor. This step ensures that participants in the control group have a similar experience to participants in the treatment group. To make things easier, there are two ways you can create this filter:
  A. Click on Settings -> Filters in Gmail -> Import filters, then upload this xml file; or
  B. Create a filter in Gmail that skips the inbox with the following values:
     To: XXX@google.com




For the next two weeks:

- The extension will prompt you to fill out a survey after you grant an LGTM to a changelist. Please fill out each survey at your earliest convenience, preferably before moving on to your next task.
- Refrain from looking at Critique messages on mobile and non-corp devices, since the extension will not be present on those devices to remove author information.
- Refrain from discussing your participation in the study with other Googlers, until the study period is complete (in about 2 weeks). In particular, authors of CLs that you review will not be aware that you are participating in this experiment, and your identity will not be blinded to the author of the CL. We will, however, contact some CL authors with a survey so that they can report if they noticed any differences in how their CLs are reviewed.
- Refrain from using Inbox during the study. The reason is that our extension does not blind Critique emails in Inbox. As a reminder, Inbox will be shutting down any day now.
- Note that our Chrome extension isn't perfect; feel free to report bugs.

In two weeks, we will send you another email informing you that the study has ended and asking you to fill out the final survey.

Emerson Murphy-Hill & Jill Dicker
Engineering Productivity Research

## 4. Screener Questionnaire

**Thanks for your interest in the study on blind code review. We define blind code review as the practice of reviewing a changelist without explicit knowledge of the author's identity.**

**<Google Data Collection Informed Consent Omitted>**

*** Required**

### If blind code review was regularly practiced at Google, I expect that...

**Significantly decreased**

**Somewhat decreased**

**No change**

**Somewhat increased**

**Significantly increased**

**I don't know**

**my engineering quality would be...**

**my engineering velocity would be...**

**my job satisfaction would be...**

**our products' ability to serve users' needs would be...**

**my engineering quality would be...**

**my engineering velocity would be...**

**my job satisfaction would be...**

**our products' ability to serve users' needs would be...**

### What do you believe the most significant benefit of blind code review will be?

Your answer

**What do you believe the most significant drawback of blind code review will be?**

Your answer

**How important is it for development tools to support blind code review at Google?**

- **Essential**
- **Worthwhile**
- **Unimportant**
- **Unwise**
- **I don't know**

*

- **I want to participate in this study.**
- **I do *NOT* want to participate in this study.**

## 5. Author Survey

When working with this CL, did you look up any information about the reviewer(s) or their work (e.g., on Teams or prior CLs)? If so, for what purpose?

During code review for this changelist, did you experience being treated as fairly as you expected?

If you encounter serious issues or potential policy violations, these issues may warrant further investigation by HR, including following up with you. You are encouraged to report concerns via any of the channels listed on go/saysomething (including the anonymous helpline open to FTEs, Interns, and TVCs which is run by a helpline provider that is entirely independent of Google) or talk to your HRBP (go/myhrbp), so it can be addressed appropriately.

- Less fairly than I expected

About the same as I expected / Don’t know / Not applicable

More fairly than I expected

Was there anything remarkable about the feedback you received from reviewers?

## 6. Post-LGTM Reviewer Form (Treatment Group)

CL Number (this answer filled in by the chrome extension)

Your answer

In your estimation, what effect did the blind code review process have, compared to normal code review?

| | Significantly decreased | Somewhat decreased | No change | Somewhat increased | Significantly increased | Don't know |
|---|---|---|---|---|---|---|
| Review quality | | | | | | |
| Review velocity | | | | | | |
| Review fairness | | | | | | |
| Review quality | | | | | | |
| Review velocity | | | | | | |
| Review fairness | | | | | | |

In this text field, the extension inserted data about your CL. This includes timestamps of when you used the unblinding feature (if at all) and when you granted LGTM. If you feel it is in error, please explain in the text box at the end of this form.

We'd like to know how often authors are guessable or otherwise known to reviewers. How certain are you that you know the identity of the author of this CL?

- Uncertain
- Somewhat certain
- Very certain

Who do you think the CL author is? (provide an LDAP)

Your answer

How did you know or guess the author's identity? (check all that apply)

- I unblinded the CL using the blind code review extension
- The author contacted me outside of Critique
- I needed to contact the author outside of Critique
- The part of the codebase being changed
- The author told me something about this change
- The nature of the change (language, programming style, etc)
- From the description of the change
- The readability status of the CL
- That this change did or didn't need OWNERS approval
- That the author did or didn't need readability approval
- I accidentally saw a notification or email about this CL
- Other:

If you unblinded the CL using the extension, why did you do so?

Your answer

If you contacted the author or the author contacted you outside of Critique, why?

Your answer

Anything else we should know about this CL?

## 7. Post-LGTM Reviewer Form (Control Group)

CL Number (this answer filled in by the chrome extension)

Your answer

In your estimation, what effect did knowing the identity of the author of this CL have, compared to not knowing their identity?

| | Significantly decreased | Somewhat decreased | No change | Somewhat increased | Significantly increased | Don't know |
|---|---|---|---|---|---|---|
| Review quality | | | | | | |
| Review velocity | | | | | | |
| Review fairness | | | | | | |
| Review quality | | | | | | |

Review velocity

Review fairness

If you contacted the author or the author contacted you outside of Critique, why?

Your answer

If you looked up any information about the author or their work (e.g., on Teams or prior CLs), for what purpose did you do so?

Your answer

Anything else we should know about this CL?

## 8. Final Questionnaire (Treatment Group)

How much did the study constraints (turning off Critique email notifications and refraining from looking at Critique emails on mobile devices) impact your productivity?

- Not at all
- Slightly impacted my productivity
- Significantly impacted my productivity

Setting aside those constraints, what effects did the blind code review process have, compared to normal code review?

Significantly decreased

Somewhat decreased

No change

Somewhat increased

Significantly increased

I don't know

Review quality

Review velocity

Review fairness

Review quality

Review velocity

Review fairness

If blind code review was regularly practiced at Google, I expect that...

Significantly decreased

Somewhat decreased

No change

Somewhat increased

Significantly increased

I don't know

my engineering quality would be...

my engineering velocity would be...

my job satisfaction would be...

our products' ability to serve users' needs would be...

my engineering quality would be...

my engineering velocity would be...

my job satisfaction would be...

our products' ability to serve users' needs would be...

What was the main benefit you experienced during this study while performing blind code review?

Your answer

What was the main drawback you experienced during this study while performing blind code review?

Your answer

To what extent were the changelists you reviewed during this study typical of those the changelists you usually review?

- Very typical
- Somewhat typical
- Not at all typical (please explain at the bottom of this form)

How important is it for development tools to support blind code review at Google?

- Essential
- Worthwhile
- Unimportant
- Unwise
- I don't know

The following questions ask about the importance of various features of a hypothetical blind code review system at Google. For each statement, please rate how important each feature would be to such a system.

Suppose a blind code review system supported blind communication between authors and reviewers, outside of Critique. How important is supporting the following types of communication?

| | Essential |
|---|---|
| Worthwhile | |

Unimportant

Unwise

I don't know

Blind chat

Blind email

Blind GVC with anonymizing face and voice distortion

Blind chat

Blind email

Blind GVC with anonymizing face and voice distortion

Suppose a blind review system is opt-in, where those who opt-in perform blind review by default, but reviewers can unblind as they see fit. How important is it to allow opt-in to be chosen by each of the following?

Essential

Worthwhile

Unimportant

Unwise

I don't know

Individual reviewers

Individual teams

Individual product areas (PAs)

The whole company

Individual reviewers

Individual teams

Individual product areas (PAs)

The whole company

During blind code review, without being informed of the author's identity, how important is it for reviewers to be able to see the following information about the author?

| | Essential | Worthwhile | Unimportant | Unwise | I don't know |
|---|---|---|---|---|---|
| Tenure at Google | | | | | |
| Level | | | | | |
| Role (SWE, SETI, UXR, etc.) | | | | | |
| Time Zone | | | | | |
| Is a TVC | | | | | |
| Is an FTE | | | | | |
| Is an intern | | | | | |
| Is on your team | | | | | |
| How many CLs you've previously reviewed from this author | | | | | |
| Whether they have readability in the language that CL is written in | | | | | |
| Tenure at Google | | | | | |
| Level | | | | | |
| Role (SWE, SETI, UXR, etc.) | | | | | |

Time Zone

Is a TVC

Is an FTE

Is an intern

Is on your team

How many CLs you've previously reviewed from this author

Whether they have readability in the language that CL is written in

## How important are the following features?

| | Essential | Worthwhile | Unimportant | Unwise | I don't know |
|---|---|---|---|---|---|
| Reviewers can unblind the author username as they see fit | | | | | |
| Critique shows the author's identity to reviewers after LGTM | | | | | |
| Critique shows which reviewers of a CL, if any, are reviewing blind | | | | | |
| Critique shows which reviewers of a CL, if any, unblinded an initially blind CL | | | | | |
| An author can request that a CL be reviewed blindly | | | | | |
| When a specific reviewer isn't required (e.g., assigned through gwsq), an author is blinded to the reviewer's identity | | | | | |
| Reviewers can unblind the author username as they see fit | | | | | |
| Critique shows the author's identity to reviewers after LGTM | | | | | |
| Critique shows which reviewers of a CL, if any, are reviewing blind | | | | | |

Critique shows which reviewers of a CL, if any, unblinded an initially blind CL

An author can request that a CL be reviewed blindly

When a specific reviewer isn't required (e.g., assigned through gwsq), an author is blinded to the reviewer's identity

Other than the features described above, what features do you think would be important to a blind code review system at Google?

Your answer

We may reach out to a small number of engineers to interview them about their experience during this study. May we contact you for an interview?

Yes

No

In this text field, the Chrome extension inserted data about the CLs you reviewed, if any data was not captured in any previous forms. This includes timestamps of when you used the unblinding feature (if at all) and when you granted LGTM. If you feel it is in error, please explain in the text box at the end of this form.

Use the text box below to provide context for your previous answers, or tell us anything else we should know about blind code review.

## 9. Final Questionnaire (Control Group)

How much did the study constraints (turning off Critique email notifications and refraining from looking at Critique emails on mobile devices) impact your productivity?

Not at all

Slightly impacted my productivity

Significantly impacted my productivity

If blind code review was regularly practiced at Google, I expect that...

Significantly decreased

Somewhat decreased

No change

Somewhat increased

Significantly increased

I don't know

my engineering quality would be...

my engineering velocity would be...

my job satisfaction would be...

our products' ability to serve users' needs would be...

my engineering quality would be...

my engineering velocity would be...

my job satisfaction would be...

our products' ability to serve users' needs would be...

To what extent were the changelists you reviewed during this study typical of those the changelists you usually review?

Very typical

Somewhat typical

Not at all typical (please explain at the bottom of this form)

How important is it for development tools to support blind code review at Google?

Essential

Worthwhile

Unimportant

Unwise

I don't know

The following questions ask about the importance of various features of a hypothetical blind code review system at Google. For each statement, please rate how important each feature would be to such a system.

Suppose a blind code review system supported blind communication between authors and reviewers, outside of Critique. How important is supporting the following types of communication?

| | Essential | Worthwhile | Unimportant | Unwise | I don't know |
|---|---|---|---|---|---|
| Blind chat | | | | | |
| Blind email | | | | | |
| Blind GVC with anonymizing face and voice distortion | | | | | |

Blind chat

Blind email

Blind GVC with anonymizing face and voice distortion

Suppose a blind review system is opt-in, where those who opt-in perform blind review by default, but reviewers can unblind as they see fit. How important is it to allow opt-in to be chosen by each of the following?

Essential

Worthwhile

Unimportant

Unwise

I don't know

Individual reviewers

Individual teams

Individual product areas (PAs)

The whole company

Individual reviewers

Individual teams

Individual product areas (PAs)

The whole company

During blind code review, without being informed of the author's identity, how important is it for reviewers to be able to see the following information about the author?

Essential

Worthwhile

Unimportant

Unwise

I don't know

Tenure at Google

Level

Role (SWE, SETI, UXR, etc.)

Time Zone

Is a TVC

Is an FTE

Is an intern

Is on your team

How many CLs you've previously reviewed from this author

Whether they have readability in the language that CL is written in

Tenure at Google

Level

Role (SWE, SETI, UXR, etc.)

Time Zone

Is a TVC

Is an FTE

Is an intern

Is on your team

How many CLs you've previously reviewed from this author

Whether they have readability in the language that CL is written in

How important are the following features?

Essential

Worthwhile

Unimportant

Unwise

I don't know

Reviewers can unblind the author username as they see fit

Critique shows the author's identity to reviewers after LGTM

Critique shows which reviewers of a CL, if any, are reviewing blind

Critique shows which reviewers of a CL, if any, unblinded an initially blind CL

An author can request that a CL be reviewed blindly

When a specific reviewer isn't required (e.g., assigned through gwsq), an author is blinded to the reviewer's identity

Reviewers can unblind the author username as they see fit

Critique shows the author's identity to reviewers after LGTM

Critique shows which reviewers of a CL, if any, are reviewing blind

Critique shows which reviewers of a CL, if any, unblinded an initially blind CL

An author can request that a CL be reviewed blindly

When a specific reviewer isn't required (e.g., assigned through gwsq), an author is blinded to the reviewer's identity

Other than the features described above, what features do you think would be important to a blind code review system at Google?

Your answer

We may reach out to a small number of engineers to interview them about their experience during this study. May we contact you for an interview?

Yes

No

In this text field, the Chrome extension may insert data about the CLs you reviewed, if any data was not captured in any previous forms. This includes timestamps of when

you used the unblinding feature (if at all) and when you granted LGTM. If you feel it is in error, please explain in the text box at the end of this form.

Use the text box below to provide context for your previous answers, or tell us anything else we should know about blind code review.

Your answer

## 10. Analysis of Miscellaneous Open-Endeds

After every CL, we asked participants: “Anything else we should know about this CL?”. Before and after the study, we also asked participants: “Use the text box below to provide context for your previous answers, or tell us anything else we should know about blind code review.”

In response to the first question, some participants provided a quick summary of the change, i.e., “C++ readability review”, “automatically generated CL”, “clean up CL”, etc. In response to both of these questions, many participants offered similar or additional context to what was requested in the other preceding open response questions within the surveys, such as how they knew the identity of the author, why they contacted them, issues with the extension, or general feedback on the concept of blind code review or requested features. There were no emergent themes beyond those reported in the analysis and summary of the other open text responses.

## 11. Manually categorized reviewer-provided reasons for inferring author identity

| Category | Description |
|---|---|
| **automated CL** | automatically generated CL |
| **bug** | Saw the author's name in a bug associated with the CL. |
| **design doc/other documentation** | Saw the author's name in a document associated with the CL, normally a design doc. In these cases the respondent did not always specify whether the document was linked within the CL description. |
| **review started before study** | The reviewer had begun reviewing the CL prior to entering the study and/or installing the extension. |
| **workspace/group name** | The workspace/group name included the author's username. |

| | |
|---|---|
| **comments in CL** | Most frequently, other reviewers addressed the author by name in their comments. Also includes cases when the author's name was included in a TODO comment. |
| **extension failed** | The extension failed to hide the author's name, often momentarily while the CL was loading. |
| **not blinded - unclear why** | The author's name was visible. Based on the level of detail in these comments it's unclear whether it was a result of the extension failing. |
| **opened on device without extension** | The reviewer opened the CL on a device on which they had not installed the blinding extension. |
| **physical proximity** | The reviewer sits close enough to the author that they saw the CL on their screen or heard them talking about it. |
| **process of elimination / list of other reviewers + CC line** | Based on the list of individuals listed as reviewers and/or CCed on the CL, the reviewer was able to deduce who the author was. Common in small teams or when the reviewer knows everyone who is working on a certain task. |
| **readability tracking spreadsheet** | The spreadsheet which assigns readability reviewers to CLs includes the author's name. |
| **task** | The reviewer knew the author was working on the content/task/featureproject associated with the CL for various reasons. Ex: heard them mention it in stand-up. See also the common sub-categories below. |
| **requested the change** | The reviewer had requested the change from the author. |
| **discussed the change beforehand** | The reviewer had discussed the change with the author before they submitted it. |
| **linked or follow up CL** | The CL in question was one within a series by the author that the reviewer was aware of and had been involved with in some way. Includes CLs following up on previous CLs. |
| **diffbase** | In several cases the author's name appeared in an associated diffbase CL. |
| **username in additional places** | The author's name showed up in an another tool which the reviewer viewed during the review. These include: git5 workspace name, sponge link in CL, g3doc preview page, code search history, screen capture. |

## 12. Guessable vs. Non-Guessable Reviews: An Analysis of Review Velocity

The analysis so far includes CLs where the reviewer knew the author's identity implicitly with those CLs where the reviewer did not. We also investigated review time, separating these two cases.

To do so, we repeat our RLR regression, with three modifications:

- We exclude readability reviews, since the number of known-author readability reviews are both exceptional and small in number.
- We exclude control group participants, which contain no unknown-author reviews by design.
- We replace the time period covariate with a condition covariate with 5 levels: Before (baseline), During-Author-Certain, During-Author-Uncertain, During-Author-Somewhat-Certain, After

This regression shows that, compared to CLs reviewed before the study began, CL reviews during the study took:

- 16% less time to review when the reviewer was uncertain of the author's identity
- 15% more time to review when the reviewer was somewhat certain of the author's identity
- 34% more time to review when the reviewer was very certain about the author's identity

All three effects were statistically significant ($p<.01$, adjusted McFadden $R^2=0.06$). This analysis suggests that greater knowledge of the author identity is associated with increased review time.

We were surprised by this finding, given that many participants told us that an experienced drawback of author anonymous code reviews was reduced velocity. However, many participants also articulated reasons why author anonymous code review actually increased their velocity. For example, participants said the main advantage of author anonymous code review was:

- "Less time trying to guess intent"
- "less time thinking about the background that the author approaches their CL with"
- "I didn't spend time considering the social dynamics of the code review interaction -- I wrote respectfully, but didn't have to fine-tune my register or my standards of review depending on my familiarity with the author."
- "taking a step ("this author may / may not know what [s]he is doing") out of the process and just try to understand the code."
- "For common CLs from other teams, I realized I didn't have to care who was sending the cl, instead I could review the cl under the assumption that they were intending (and should be intending) to do what the cl description stated."
- "No time wasting looking up the author and their team"

Nonetheless, we were skeptical that knowledge of author identity alone could increase reviewing speed so substantially. To investigate further, we modified our regression model above to predict the number of comments and average comment length in a CL. These models revealed that being uncertain about author identity correlated with fewer comments (-18%, $p<.001$, adjusted McFadden $R^2=0.10$) and shorter average comment length (-12%, $p<.001$, adjusted McFadden $R^2=0.006$). As a whole, we interpret this to mean that uncertain participants reviewed code faster largely because, at least in part, they were making fewer and shorter comments.

But did author anonymous code reviewers write fewer and shorter comments because they didn't know the author's identity, or because CLs with unknown authors are fundamentally different from those CLs with known authors?

To investigate this question, we directly asked participants about why they "might have added fewer comments on these 'uncertain' CLs?". In particular, we sent follow up emails to the participants who reported at least 10 instances of being uncertain about author identities. Fourteen out of 15 participants we emailed responded, citing the following rationales:

- Five participants said that trivial or straightforward changes -- small modifications, library upgrades, clean ups, and so on -- could both be performed by multiple potential authors and are unlikely to incur code review comments.
- Two participants noted that their roles as Site Reliability Engineers meant that they often reviewed configuration changes from people they did not know, changes which had minimal impact on production so were unlikely to incur comments.
- Two participants noted that they reviewed a large number of CLs that are largely boilerplate, which both get authored by people they don't know and are unlikely to contain problems worth commenting on. Similarly, another participant said that many CLs they review are created by a variety of engineers using a tool specific to his org, a tool whose changes typically don't require much feedback.
- One participant noted that the more comments on a CL, the more likely the discussion about the CL would be taken offline, breaking anonymity.
- One of the participants who reviews a lot of changes submitted by different teams said that in the cases where they are certain of author identity, they have been more involved with the project beforehand and have stronger interest, opinions, and feedback to offer, which results in more comments.
- One participant said they could determine a CL author's identity by the way they responded to comments, so the more comments they left, the easier it was to determine their identity.
- One participant said that CLs from people he doesn't know are likely to get fewer comments from him, because the participant also does not know the codebase well.
- One participant noted that their team recently grew substantially to include many people they did not know, and the participant performed code reviews for the new people specifically for the purpose of increasing his knowledge of those people's code, rather than to provide feedback.

- One participant said that knowing the author's identity allows them to leave more tailored feedback, which in some cases, would result in more comments (e.g., for a new employee).

This last comment was the only one where the reviewer said they wrote fewer comments because they didn't know the author's identity. Based on the other comments, we conclude that CLs with unknown authors are often fundamentally different from those CLs with known authors.

## 13. Hypotheses for Why Control Group Authors Perceived More Fair for Readability Reviews

- Perhaps the results are a statistical anomaly caused by a violation of the model's assumption of a linear relationship between the three outcomes (more fair, less fair, same). To test this hypothesis, we ran two separate linear regressions with the same independent variables a binary dependent variable: Model A that combined the "less fair" and "same" responses and Model B that combined the "more fair" and "same" responses. Model A confirmed our original model, that for readability, control group participants were more likely to receive a "more fair" rating. Model B showed no effect of group, that control and treatment groups did not show a significant difference in likelihood to report a less fair experience than expected. These results suggest that our original model's differences were driven largely by the "more fair" ratings, and that our results are unlikely to be caused by violating the linearity assumption.
- Perhaps a few reviewers were disproportionately contributing multiple high-fairness reviews from multiple authors. This hypothesis is bolstered by the fact that of the 14 reviewers in the treatment group to receive a "more fair" review rating, only 1 received multiple "more fair" ratings; in the control group, 5 of 13 did. To investigate this hypothesis, we ran a second regression predicting mean fairness scores per-reviewer using a similar regression to the one described above. This model also showed that readability reviews given by reviewers in the control group were *still* more likely to give "more fair" ratings. So this hypothesis is also not confirmed.
- Perhaps during author anonymous code review, authors at lower levels may feel more fairly treated while reviewers at higher levels may feel less fairly treated having previously been granted more leeway during code review, leeway that was not granted during author anonymous review. This hypothesis would explain our results above if these two opposite effects were cancelling each other out. To investigate this, in our regression models we included a 3-way interaction between control/treatment, readability, and author level. No significant interaction emerged, disconfirming this hypothesis.